HOW TO BUILD 250 SINGLE PAGE APPLICATIONS WITH AWS LAMBDAS

WHO AM I ?

NAME: Max [1]

SURNAME: Gallo

TWITTER: @_maxgallo

SPECS: Biped, Pasta eater, Work at DAZN, JavaScript developer, mediocre iOS Developer, Functional & Reactive programming learner

[1] or Massimiliano if you like Italian spelling challenges

WHY AM I TALKING IN THE AWS USER GROUP UK ?

Because since March I've been working with AWS Lambdas, S3 and DDB

TO CREATE THE DAZN 2.0
Front End BUILD PIPELINE.

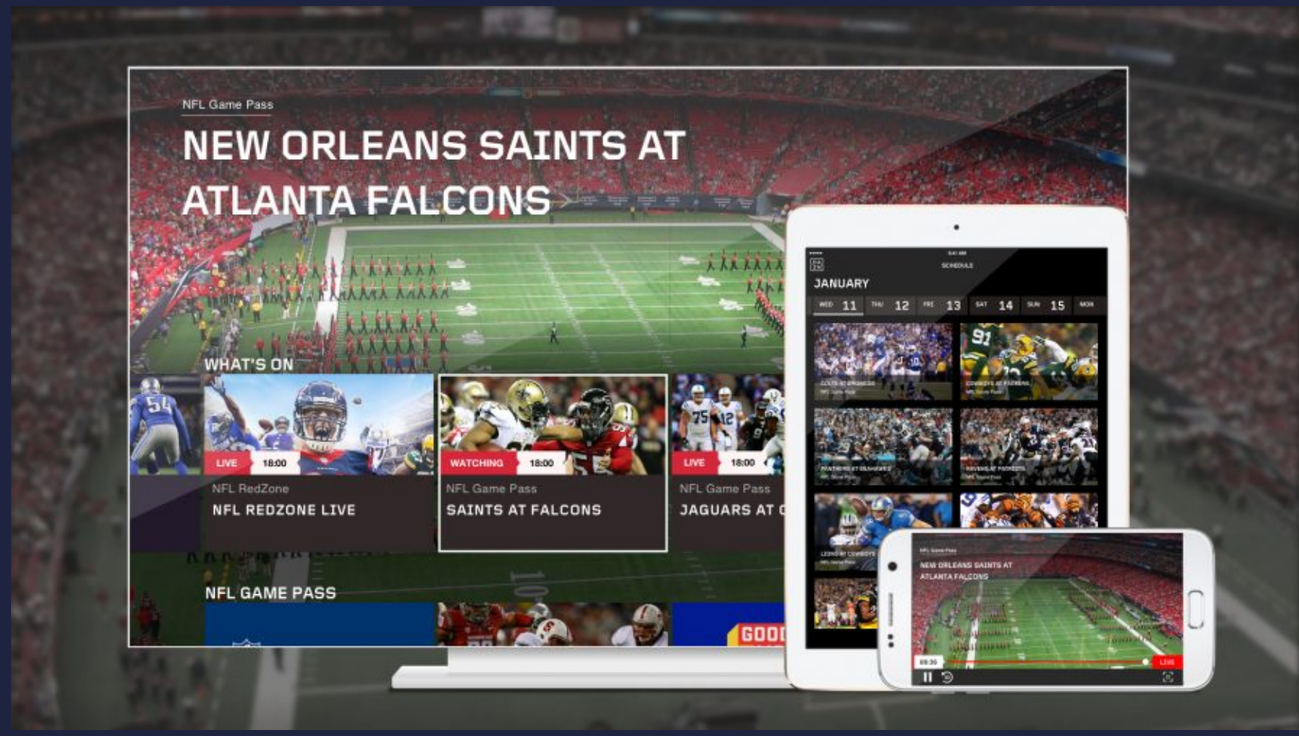
WHAT I'M GOING TO *talk* ABOUT

- Introduction to DAZN
- Why we needed a Build Pipeline
- How it works
- Takeaways

DAZN

THE NETFLIX OF SPORT

- Monthly subscription
- All Sports Included
- Available on any device



DAZN 2.0

MULTIPLE *Single Page Applications*, TAILORED PER COUNTRY AND DEVICE

HOW MANY SPA ARE WE TALKING ABOUT?

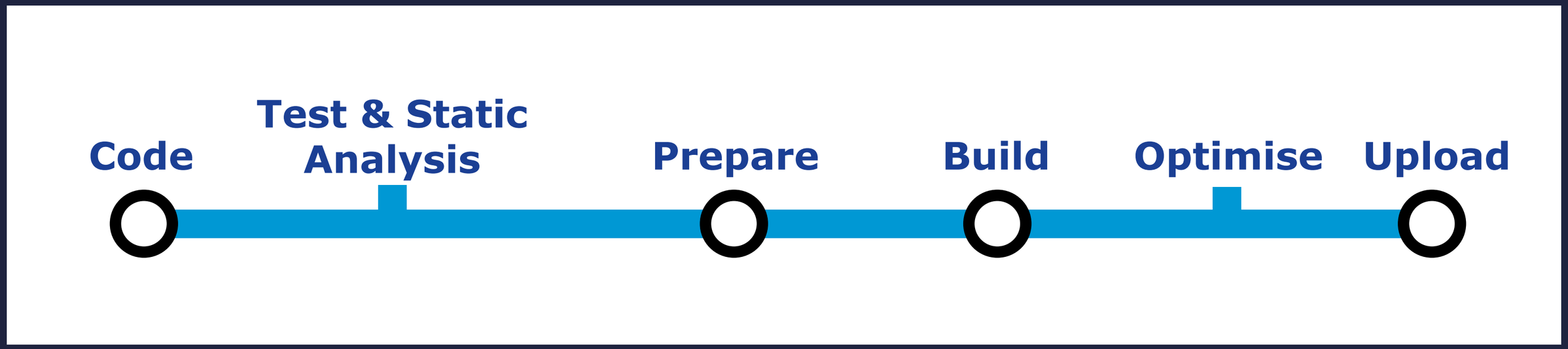
HOW MANY	WHAT	SERIOUSLY, WHAT?
1	Application	dazn.com
5	Countries	JP, DE, CA, CH, AT
5	Chapters	catalog, auth, help, ...
10	Targets	Web, Mobile, Android Tv, Xbox, Amazon Firestick, ...

We *just* need to build 5 * 5 * 10 = 250 SPA

WHAT WE WANT TO ACHIEVE

- Easy to maintain and adopt
- Not always on - It's a build system after all
- Infrastructure as code

WELCOME TO: *The Tube*

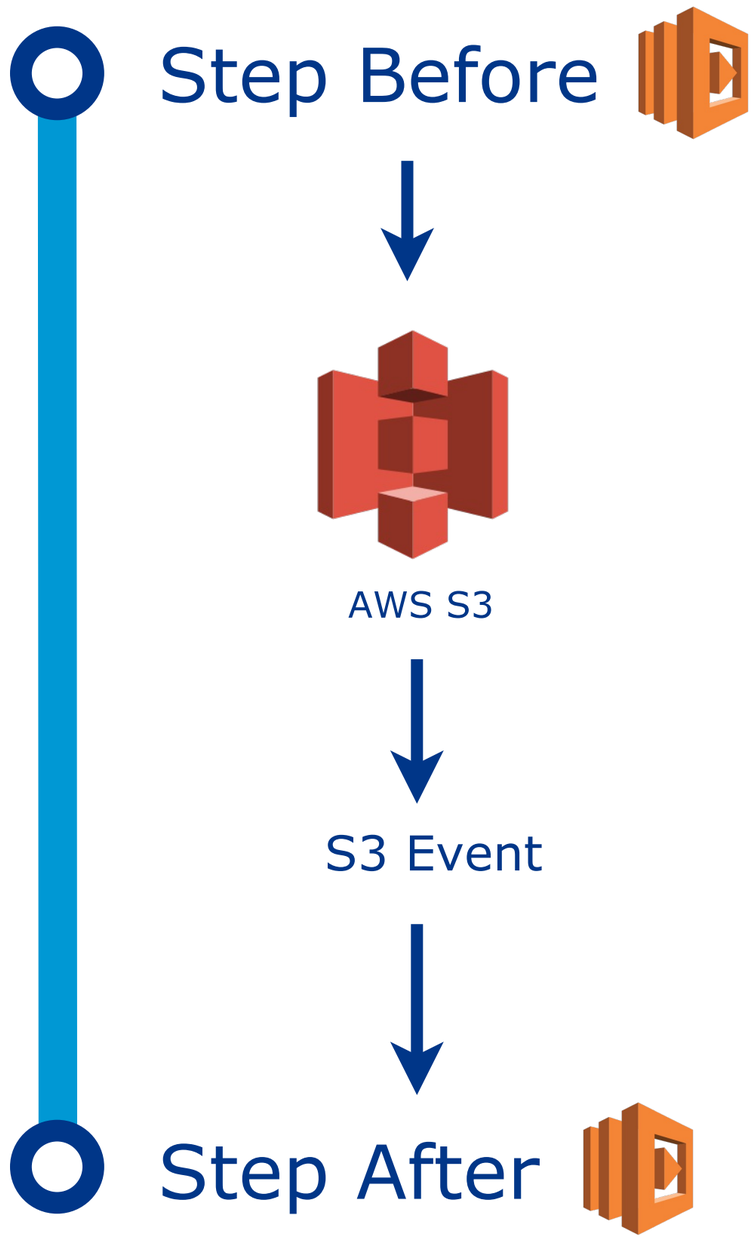


MOVING IN THE *Tube*

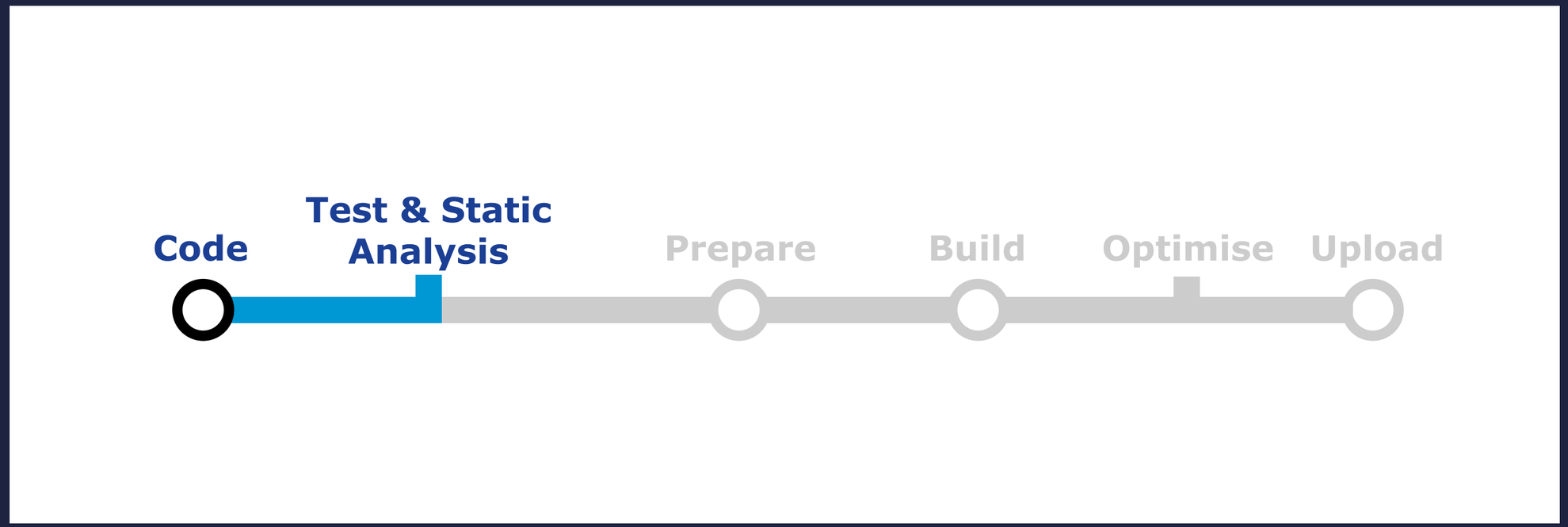
FUNCTIONAL & REACTIVE

- Each [2] Station is an AWS Lambda
- Each Station is Stateless and Atomic
- Each Station is triggered by event reaction

[2] almost each



FROM Code TO Test & Static Analysis



INSIDE *Code*

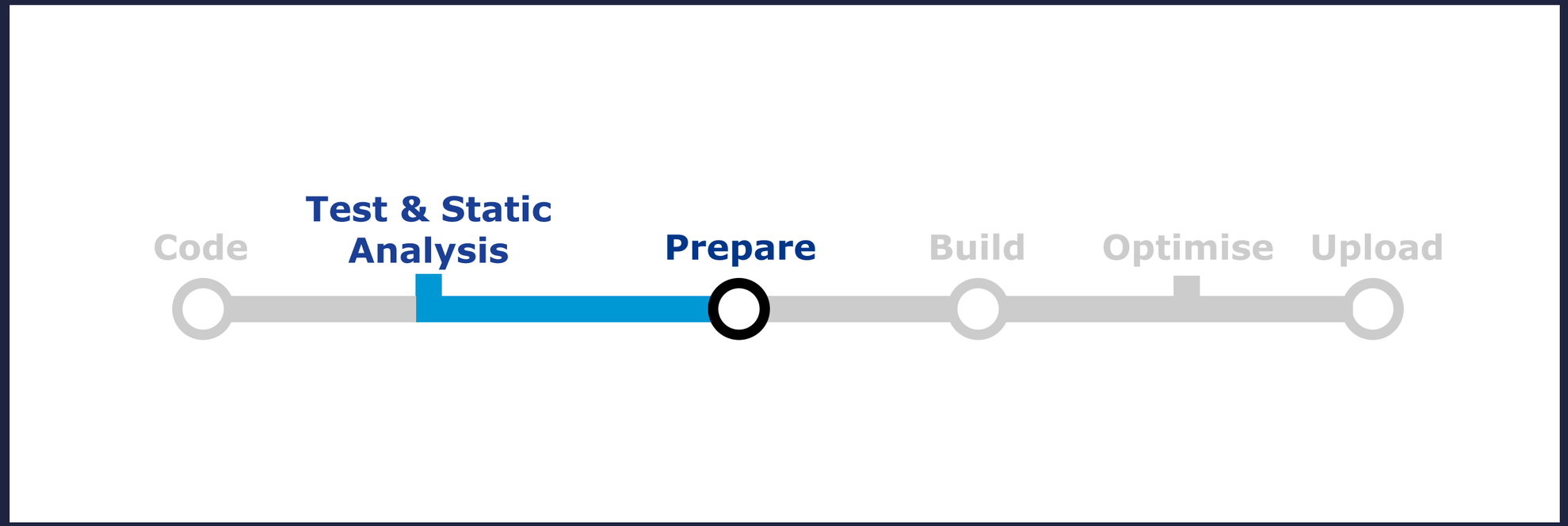
- One repo per Target
- Each Target Repo contains all the chapters
- GitHub webhook to a Lambda functions to trigger the next step

INSIDE Test & Static Analysis

Custom implementation, using Docker Swarm and Bash

1. Checkout target project from GitHub
2. Run Unit Tests
3. Run Static Analysis
4. Compress everything and store on S3

FROM Test & Static Analysis TO Prepare

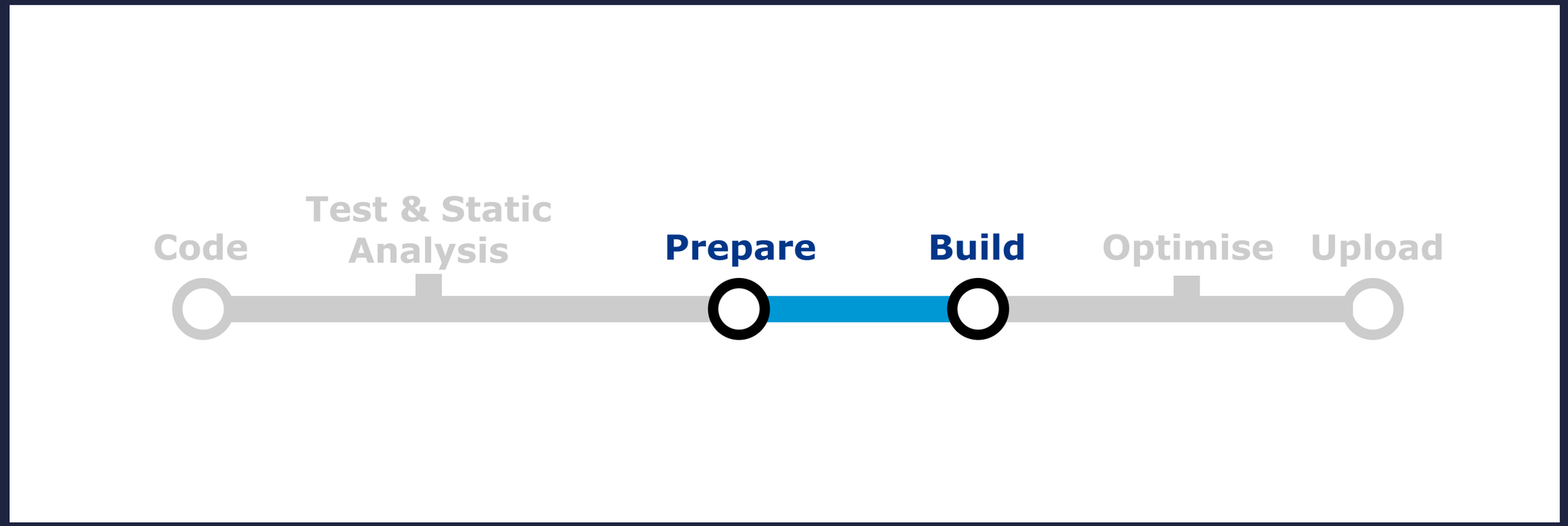


INSIDE *Prepare*

1. Download the project code from S3
2. Magic [3]
3. Pack everything and re-upload to S3

[3] You have to trust me on this one

FROM Prepare TO Build

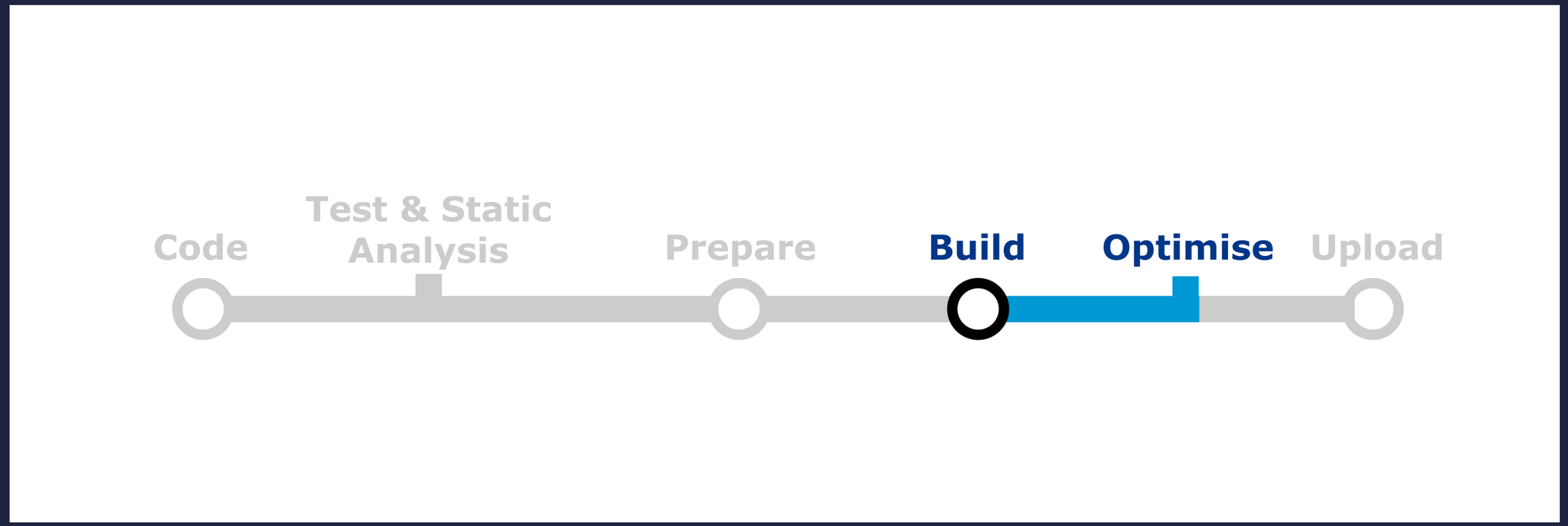


INSIDE *Build* LAMBDA

IT BUILDS THE PROJECT CODE, WITHOUT KNOWING HOW TO BUILD IT.

1. Download the project code from S3
2. Read configurations from the project code
3. Build the project
4. Pack the build output files and upload to S3

FROM Build TO Optimise

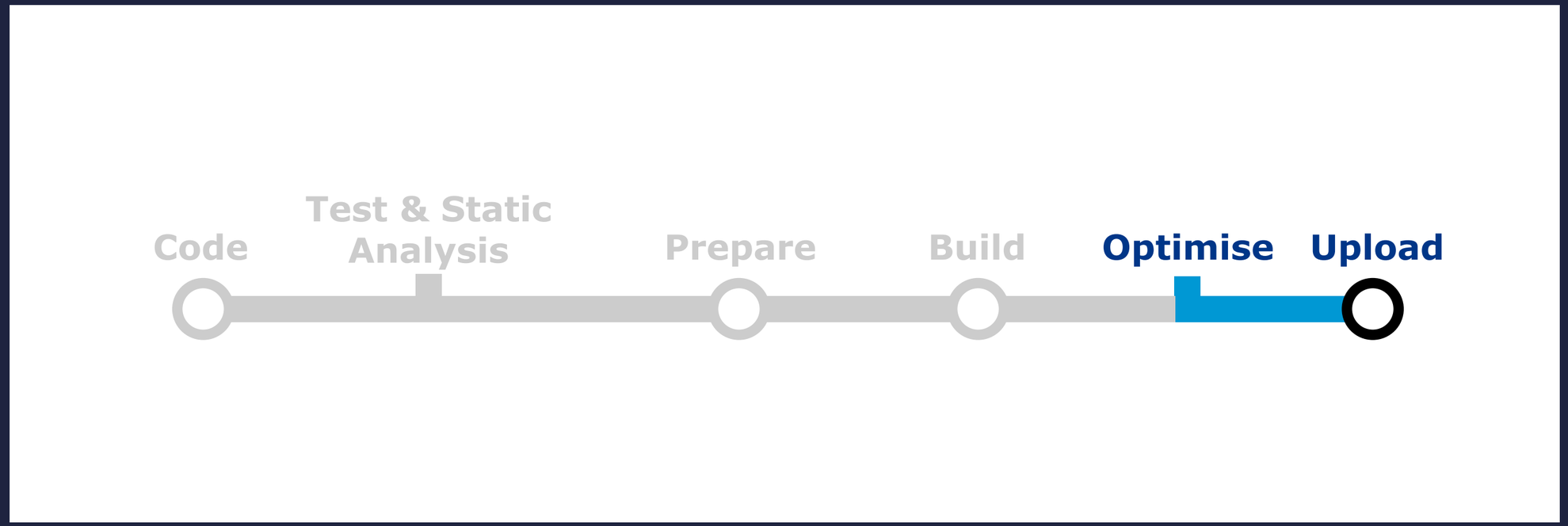


INSIDE *Optimise* LAMBDA

OPTIMISE THE BUILD OUTPUT.

1. Download the build output files from S3
2. Optimise JavaScript files
3. Optimise Html and CSS as well.
4. Pack the optimised files and upload them to S3
5. This code is now ready to be used

FROM Optimise TO Upload

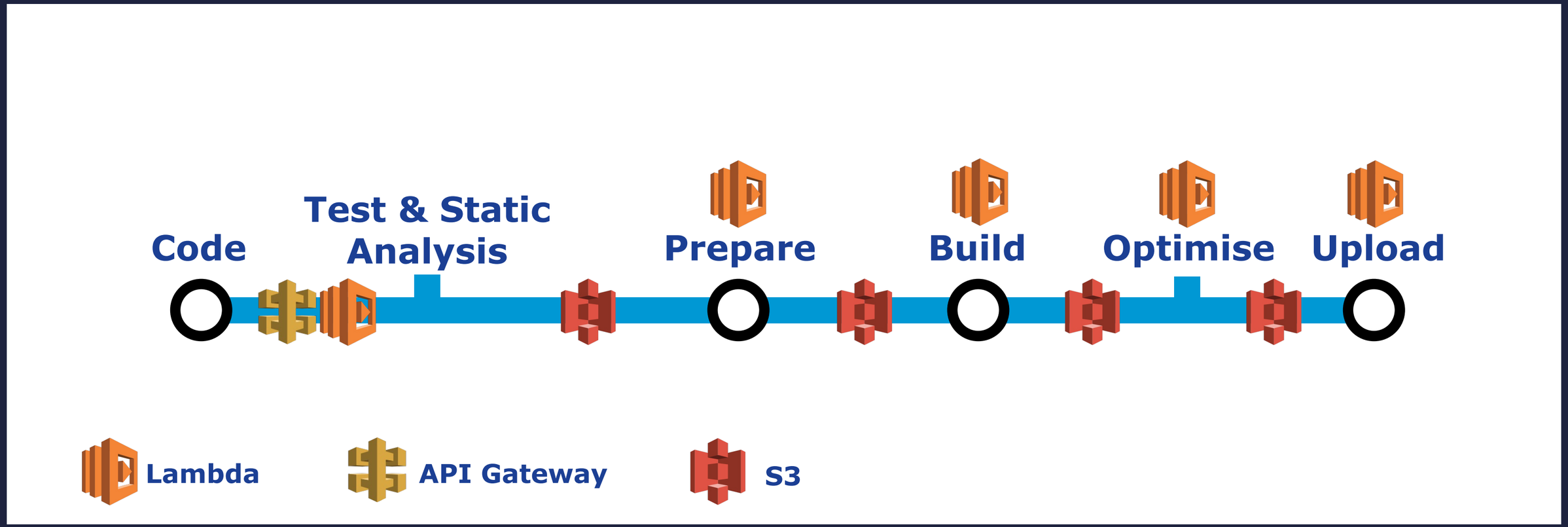


INSIDE *Upload* LAMBDA

UPLOAD THE BUILD OUTPUT.

1. Download the optimised build output files from S3
2. Upload the code to Artifactory
3. Update Database about new available build

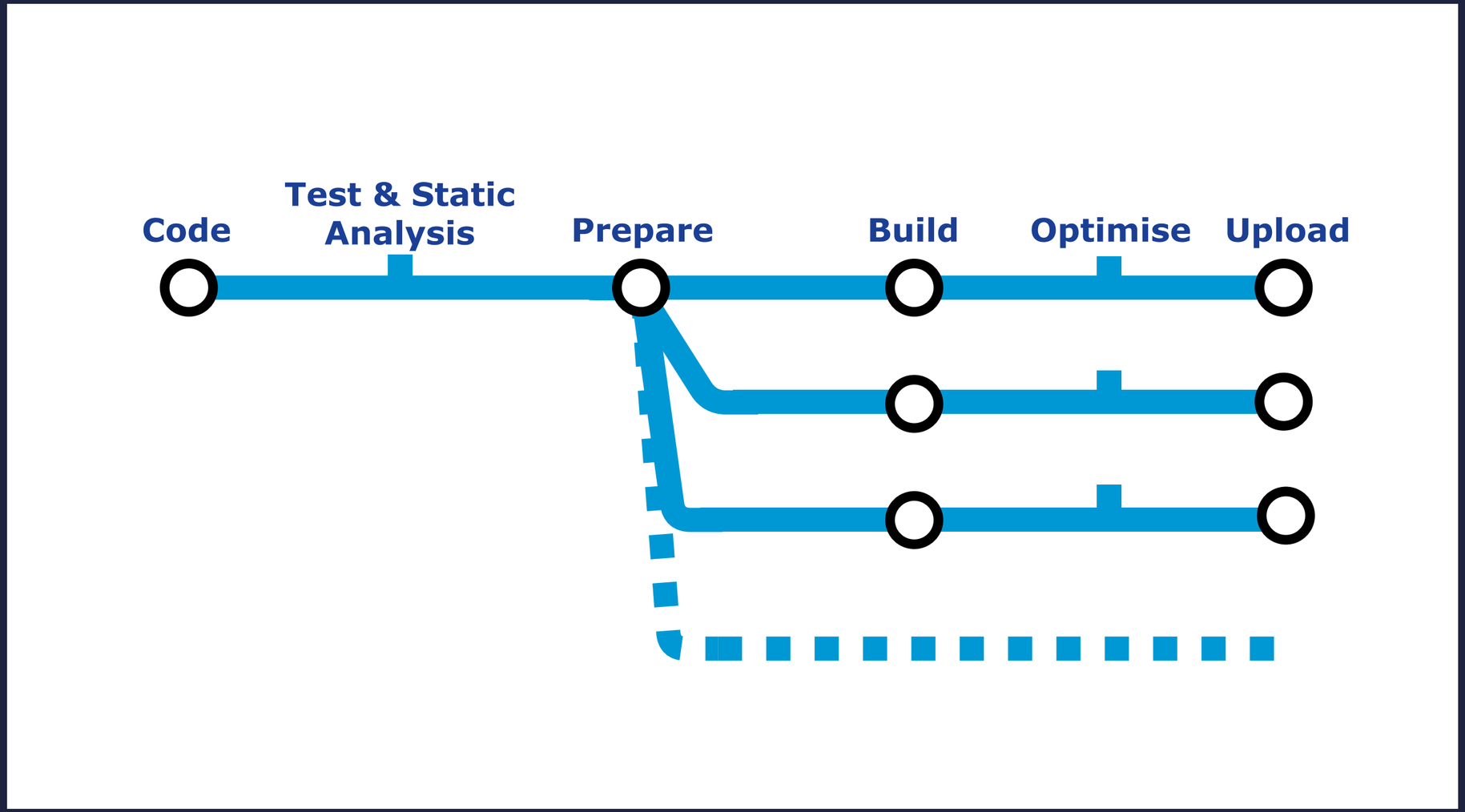
PUTTING ALL TOGETHER



WAIT !

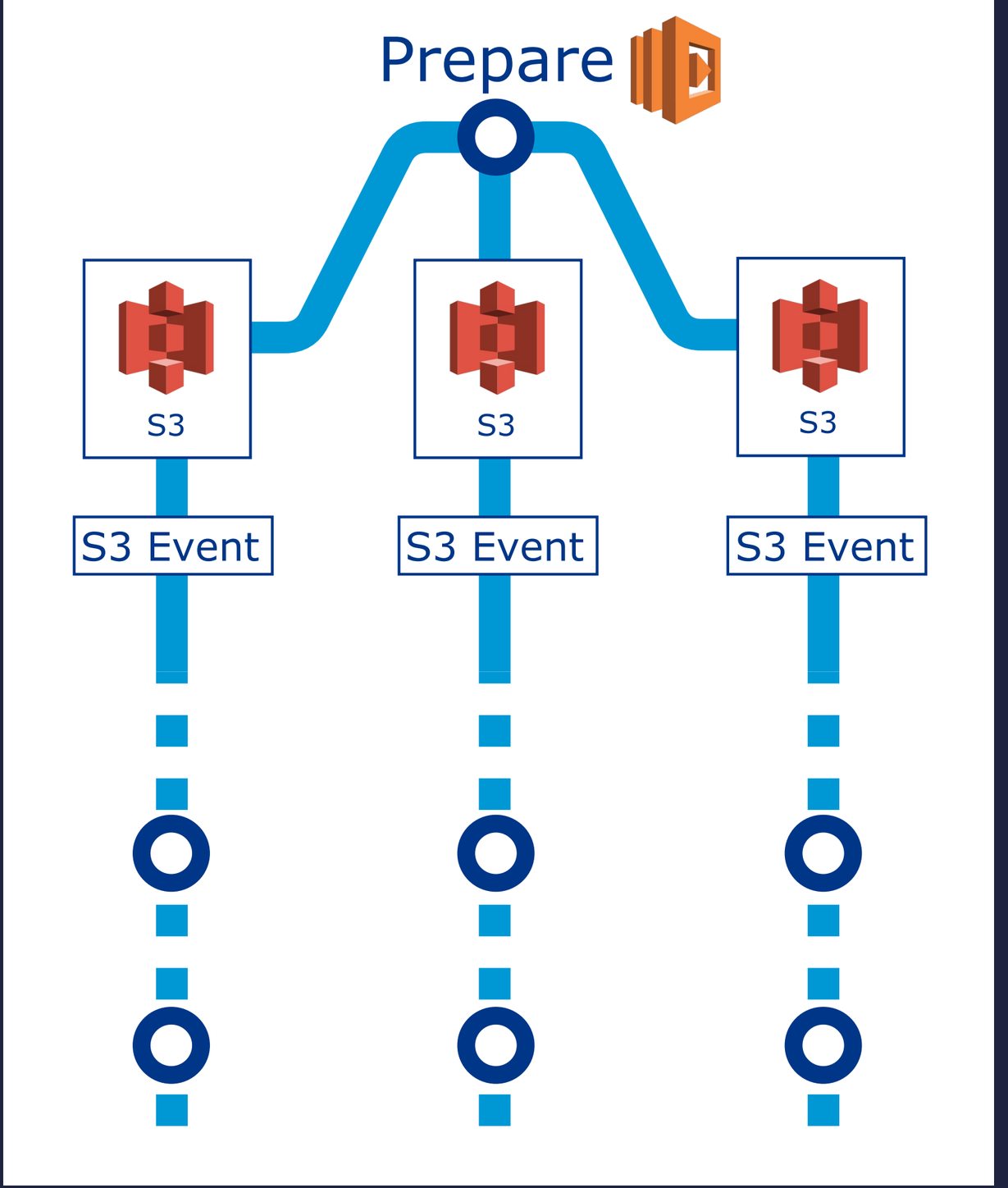
What about the 250 Single Page Applications ?

Truth is, that the *Tube* is more like this



Prepare IS ACTUALLY A MULTIPLIER

- It saves multiple files on S3
- Multiple events generated
- Multiple lambdas triggered



INSTEAD OF ONE PIPELINE

WE NOW HAVE 250 OF THEM!

TAKEAWAYS

ORGANISATIONAL & TECHNICAL

ORGANISATIONAL

- Easy learning curve
  - + Devs - Ops
- Relatively low costs
- You can use different Cloud Vendors

TECH

- Frameworks & Language agnostic
- Bounded Context
- Reactive Flow

THANKS

twitter: @_maxgallo

slides: github.com/maxgallo/lambdas-in-dazn

medium: https://medium.com/dazn-tech